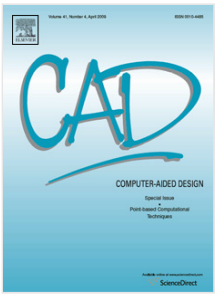
Accepted Manuscript



Three dimensional extension of Bresenham's algorithm with Voronoi diagram

Chikit Au, Tony Woo

PII: S0010-4485(10)00216-2
DOI: 10.1016/j.cad.2010.11.006
Reference: JCAD 1690

To appear in: *Computer-Aided Design*

Received date: 14 January 2010
Accepted date: 18 November 2010

Please cite this article as: Au C, Woo T. Three dimensional extension of Bresenham's algorithm with Voronoi diagram. *Computer-Aided Design* (2010), doi:10.1016/j.cad.2010.11.006

JCAD_1690

- The efficiency of Bresenham algorithm for plotting a 2D line is examined.
- An extension of the algorithm for representing a 3D line is proposed.
- The efficiency of the algorithm is based on the space symmetry.
- It is further improved by setting a simple grid point hierarchy and Voronoi diagram.

Three Dimensional Extension of Bresenham's Algorithm with Voronoi Diagram

Chikit Au*
Department of Engineering
University of Waikato, New Zealand

Tony Woo
School of Mechanical and Aerospace Engineering
Nanyang Technological University, Singapore

Abstract

Bresenham's Algorithm for plotting a two-dimensional line segment is elegant and efficient in its deployment of mid-point comparison and integer arithmetic. It is natural to investigate its three-dimensional extensions. In so doing, this paper uncovers the reason for little prior work. The concept of the mid-point in a unit interval generalizes to that of nearest neighbours involving a Voronoi diagram. Algorithmically, there are challenges. While a unit interval in two-dimension becomes a unit square in three-dimension, "squaring" the number of choices in Bresenham's Algorithm is shown to have difficulties. In this paper, the three-dimensional extension is based on the main idea of Bresenham's Algorithm of minimum distance between the line and the grid points. The structure of the Voronoi diagram is presented for grid points to which the line may be approximated. The deployment of integer arithmetic and symmetry for the three-dimensional extension of the algorithm to raise the computation efficiency are also investigated.

Keywords: Voronoi Diagram, Bresenham Algorithm, integer arithmetic, symmetry

* Corresponding Author: C. K. AU (ckau@waikato.ac.nz)
Department of Engineering, University of Waikato, New Zealand

1. Introduction

Bresenham's algorithm[1] is efficient in generating straight lines and quadrics on a raster system. Furthermore, its concept has a wide range of applications such as re-sampling of structured grids[2], line of sight calculation between a sensor and a target[3], interpolation in computer numerical control systems[4], ray casting[5], ray tracing[6], volume rendering[7], three-dimensional map representation[8], occlusion checking in re-constructing a three dimensional object[9, 10], navigation for autonomous flight[10] and collision detection[11].

All these applications involve casting a line between two points and employ a "half and half" approach to extend the Bresenham's Algorithm in a three-dimensional space. A line is sampled by a set of points in the three-dimensional space with a grid. The grid points are selected with respect to the sample points to represent the line. Figure 1 shows the "half and half" grid point selection process. One of the coordinate directions (assume *y* coordinate as shown in figure 1) is used as a driving axis and the line is sampled along that direction. {$\mathbf{P}_i$, $\mathbf{Q}_i$, $\mathbf{R}_i$, $\mathbf{S}_i$} and {$\mathbf{P}_j$, $\mathbf{Q}_j$, $\mathbf{R}_j$, $\mathbf{S}_j$} are two sets of grid points. Point $\mathbf{T}_i$ and $\mathbf{T}_j$ are two sample points from the line. Since $\mathbf{T}_i$, $\mathbf{P}_i$, $\mathbf{Q}_i$, $\mathbf{R}_i$ and $\mathbf{S}_i$ are on the same plane so that they have the same *y* coordinate. The sample point $\mathbf{T}_i$ is then projected onto the edges $\mathbf{P}_i\mathbf{S}_i$ and $\mathbf{P}_i\mathbf{Q}_i$. The *x* and *z* coordinates of the grid points, which are closest to the projected sample point, will be adopted. Hence, the grid point $\mathbf{S}_i$ and $\mathbf{R}_j$ will be selected in this example. Obviously, the error is faced with a threshold of $\frac{1}{2}$ for each coordinate.

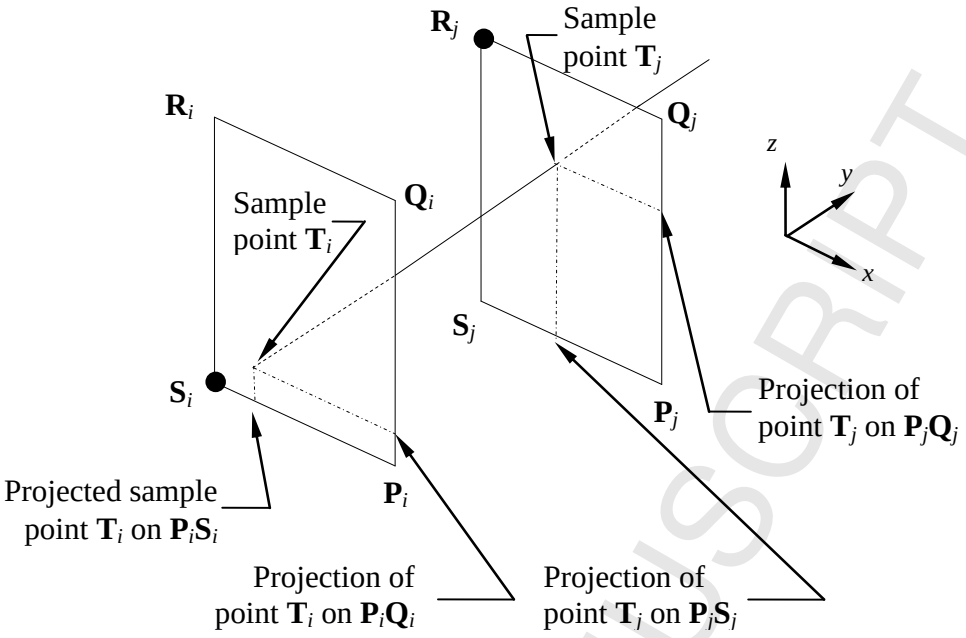


Figure 1. The "half and half" approach for 3D Bresenham algorithm

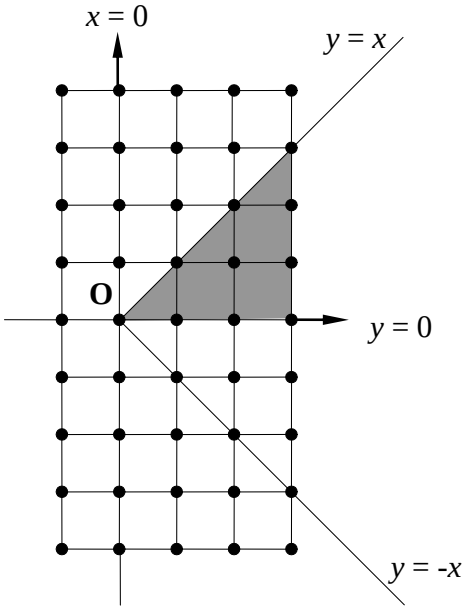
This three-dimensional extension is equally efficient as the Bresenham's algorithm. Both approaches employ the same mechanism to select the appropriate grid points with a threshold error of $\frac{1}{2}$. Using this threshold error for two-dimensional implementation is natural. However, there is no evidence to show that threshold error of $\frac{1}{2}$ is equally applicable in the three-dimensional implementation. In fact, this heuristic approach does not select the appropriate grid points to properly represent a straight line in a three-dimensional space. This paper discusses a grid point selection mechanism to represent a straight line in a three-dimensional space. An algorithm, which follows the Bresenham's Algorithm approach, is developed. A Voronoi diagram is established to partition the space with respect to the grid. The grid points, which possess the minimum distance from the line, are selected based on the membership of the sample point. An example is raised to compare the proposed algorithm and the current implementation.

2. Bresenham's Algorithm

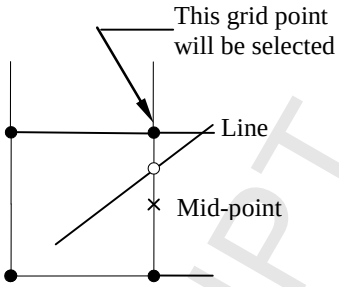
Bresenham's Algorithm is efficient at selecting a set of grid points to represent a straight line in a two-dimensional space. Space symmetry and computation are two major aspects for the algorithm to earn efficiency.

2.1 Symmetry in two-dimensional Space

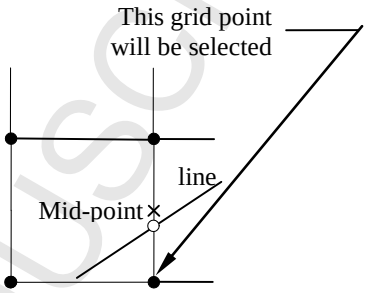
The Bresenham's algorithm maps a given line segment to the first octant, beginning at (0, 0) and ending at (*a*, *b*) by exploiting eight-fold symmetry (on octants). Figure 2(a) shows a two-dimensional plane. The shaded area is an octant of the plane with a rectangular array of grid points. A line with positive slope less than one is mapped to the first octant for grid point selection. A set of sample points, which is shown as a white dot in the figure, on the line is generated along the *x* axis with unit interval. The algorithm determines a sequence of grid points with an error no greater than $\frac{1}{2}$ as shown in figure 2(b) and 2(c). For a line segment with slope greater than one, the roles of the *x* and *y* coordinates in the algorithm swap. Hence, the sample points are generated along the *y* axis. Furthermore, the grid point selections must be consistent when a sample point is in the middle of two consecutive grid points so that the same sequence of grid points are selected regardless of the starting end point. For a line with negative slope, the algorithm is modified so that one coordinate decreases and the other increases. Finally, the vertical line, diagonal line and horizontal line are all considered as special cases to handle. A detailed explanation of the implementation can be found in reference 12.



(a) An octant (shaded area)



(b) The upper grid point is selected



(c) The lower grid point is selected

Figure 2. The octant and the grid point selection mechanism

2.2 Computation Efficiency

There are two key ideas in Bresenham's algorithm computation, one for computing the distance efficiently and the other for enabling integer arithmetic. They are inter-related.

The first idea involves converting the perpendicular distance between a point and a line to the *y*-intercept of the line for mid-point comparison with the value $\frac{1}{2}$. Figure 3 illustrates the selection rule for the two choices **P** and **Q**, with the corresponding perpendiculars **P**' and **Q**' on the line:

Two points and two magnitudes: if $|\mathbf{PP}'| < |\mathbf{QQ}'|$ then choose **P**

else choose **Q**

Rather than calculating the magnitudes $|\mathbf{PP'}|$ and $|\mathbf{QQ'}|$, the *y*-intercept, which involves only one point **T**, is computed. In particular, $\frac{|\mathbf{PP'}|}{|\mathbf{PT}|} = \frac{|\mathbf{QQ'}|}{|\mathbf{QT}|}$ since $\Delta\mathbf{PP'T}$ and $\Delta\mathbf{QQ'T}$ are similar. The selection rule becomes:

One point and two *y*-intercepts: if $|\mathbf{PT}| < |\mathbf{QT}|$ then choose **P**

else choose **Q**

Assuming $|\mathbf{PT}| + |\mathbf{TQ}| = 1$ unit of grid spacing, a further improvement in the efficiency can be made by eliminating one of the two *y*-intercepts.

One point and one *y*-intercept: if $|\mathbf{PT}| < ½$ then choose **P**

else choose **Q**

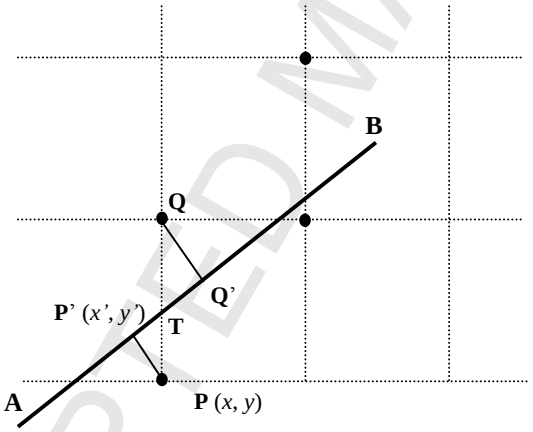


Figure 3. Perpendicular distances as *y*-intercept due to similarity in $\Delta\mathbf{PP'T}$ and $\Delta\mathbf{QQ'T}$

The above improvements also reduce the arithmetic complexity in the operations. The coordinates of point **P**'(*x*',*y*') adopts a parametric value *u* in the linear equation for the line from **A**(0,0) to **B**(*a*,*b*) such that (*x*', *y*')=(*ua*, *ub*). The parametric value *u* is expressed as

$u = \dfrac{ax + by}{\sqrt{a^2 + b^2}}$. By contrast, finding the *y*-intercepts **T** of a line from (0, 0) to (*a*, *b*) at the various *x*-coordinate x_i involves only: $y_i = \dfrac{b}{a} x_i$.

This leads to the second idea in Bresenham's Algorithm of using integer arithmetic. A glance at the assignment statement $y_i = \dfrac{b}{a} x_i$ reveals that x_i on the right hand side are integer. But the *y*-intercept on the left-hand side must necessarily be a real number. Revisiting the one point and one y-intercept statement: "if $|\mathbf{PT}| < ½$ then choose **P**" provides a solution to enable the integer arithmetic. The right-hand side is converted to an integer by multiplying the left hand side of the comparison by 2 (which amounts to a left-shift by 1 bit) then

One point and one *y*-intercept:	if	$2\|\mathbf{PT}\| < 1$	then choose **P**
	else		choose **Q**

Rather than comparing the magnitude $|\mathbf{PT}|$ with $\dfrac{1}{2}$, scaling the difference $|\mathbf{PT}|$ by 2 (corresponding to a left-shift) allows comparison with the integer 1. In implementation, twice the magnitude $|\mathbf{PT}|$ is stored in an accumulator ∇ which overflows when it exceeds 1.

Bresenham's Algorithm between two points **A**(0,0) and **B**(*a*,*b*) in the first octant is now ready for presentation in pseudo-code.

```
Procedure BresenhamLine { A(0, 0) B(a, b) }
x   = a                                          ; initialization
y   = b
∇0   = 2 y – x
for i = 0 to a   do begin
     if   ∇i < 0   then    choose P
                           ∇i+1 =   ∇i + 2 y          ; update difference
                    else   choose Q
                           ∇i+1 =   ∇i + 2 y – 2 x    ; update difference
end-do
EndProcedure
```

3. Voronoi Diagram

A Voronoi digram[13] shows the partitioning of a two-dimensional space $\Re^2$. Each partition is determined by distances to a specific discrete set of points which are called sites. Two Voronoi regions $R(\mathbf{P}, \mathbf{Q})$ and $R(\mathbf{Q}, \mathbf{P})$ are defined between any two sites **P** and **Q** ($\mathbf{Q} \neq \mathbf{P}$) as,

$$R(\mathbf{P},\mathbf{Q}) = \left\{\mathbf{D} \mid dist(\mathbf{D},\mathbf{P}) < dist(\mathbf{D},\mathbf{Q}), \mathbf{P} \in \Re^2, \mathbf{Q} \in \Re^2\right\} \quad (1a)$$

$$R(\mathbf{Q},\mathbf{P}) = \left\{\mathbf{D} \mid dist(\mathbf{D},\mathbf{Q}) < dist(\mathbf{D},\mathbf{P}), \mathbf{P} \in \Re^2, \mathbf{Q} \in \Re^2\right\} \quad (1b)$$

where $dist(\mathbf{D},\mathbf{P})$ and $dist(\mathbf{D},\mathbf{Q})$ are distance functions of the point **D** from the sites **P** and **Q** respectively. Therefore, two regions are defined for every two sites and

$$R(\mathbf{Q},\mathbf{P}) \cup R(\mathbf{P},\mathbf{Q}) = \Re^2. \quad (2)$$

A Voronoi curve is defined between two Voronoi regions

$$\partial R(\mathbf{P},\mathbf{Q}) = \left\{\mathbf{D} \mid dist(\mathbf{D},\mathbf{P}) = dist(\mathbf{D},\mathbf{Q})\right\} \quad (3)$$

Hence, $\partial R(\mathbf{P},\mathbf{Q}) = \partial R(\mathbf{Q},\mathbf{P})$.

A Voronoi diagram $V(\mathbf{G})$ of a given set **G** of sites consists of the set of Voronoi cells $V(\mathbf{P})$ ($\forall \mathbf{P} \in \mathbf{G}$) which is the boolean intersection of all the Voronoi regions $R(\mathbf{P},\mathbf{Q})$ $\forall \mathbf{Q} \neq \mathbf{P}$ and $\forall \mathbf{Q} \in \mathbf{G}$.

$$V(\mathbf{P}) = \bigcap R(\mathbf{P},\mathbf{Q}), \quad \mathbf{P} \neq \mathbf{Q} \quad \text{and} \quad \forall \mathbf{Q} \in \mathbf{G} \tag{4}$$

Hence, a Voronoi cell $V(\mathbf{P})$ of a specific site **P** is given as

$$V(\mathbf{P}) = \{\mathbf{T} \mid dist(\mathbf{T},\mathbf{P}) < dist(\mathbf{T},\mathbf{Q}), \mathbf{P} \neq \mathbf{Q}, \mathbf{P} \in \mathbf{G}, \forall \mathbf{Q} \in \mathbf{G}\} \tag{5}$$

The boundary of a Voronoi cell $\partial V(\mathbf{P})$ is composed of various segments of Voronoi curves $\partial R(\mathbf{P},\mathbf{Q})$ ($\forall \mathbf{Q} \neq \mathbf{P}$ and $\forall \mathbf{Q} \in \mathbf{G}$).

4. Extension of Bresenham's Algorithm to Three-dimensions

Similar to sampling a two-dimensional line by *y*-intercept, a three-dimensional line is sampled by a set of intersection points as it pierces a series of planes (parallel to one of the principal planes) through the grid points. The grid points which are nearest to the line are selected to represent a line in three-dimensional space.

Figure 4 shows a line segment **AB** originating from the origin **A**(0, 0, 0) and making an angle with a square **PQRS** (which are the grid points). Let the coordinates of the point of intersection **T** (which is a sample point) between the given line segment and a square be (*x*, *v*, *w*) and the foot of the perpendicular of **P**, **Q**, **R** and **S** onto the line be **P**', **Q**', **R**' and **S**' respectively (only **P**' is shown in figure 4).

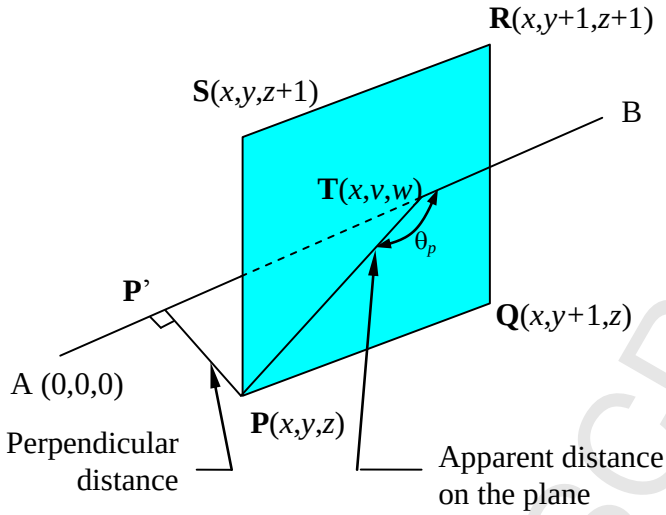


Figure 4. A line passes through a plane

The perpendicular distance from **P** to the line segment **AB** is $|\mathbf{TP}|\sin\theta_p$, where θ_p is the angle of intersection given by:

$$\cos\theta_p = \frac{v(y-v)+w(z-w)}{\sqrt{(x^2+v^2+w^2)\left((y-v)^2+(z-w)^2\right)}} \tag{6}$$

Similarly, there are three other perpendicular distances: $|\mathbf{TQ}|\sin\theta_q$, $|\mathbf{TR}|\sin\theta_r$ and $|\mathbf{TS}|\sin\theta_s$ from the other three grid points **Q**, **R** and **S**, respectively. The expressions of the angle θ_q, θ_r and θ_s are listed in Appendix A.

A Voronoi diagram can be generated on the square **PQRS** to determine the membership of the sample point **T** by defining the distance function as:

$$dist(\mathbf{TD}) = |\mathbf{TD}| \cdot \sin\theta_D, \qquad \forall \mathbf{D} \in \{\mathbf{P}, \mathbf{Q}, \mathbf{R}, \mathbf{S}\} \tag{7}$$

Because the grid point with the shortest perpendicular distance to the line is the best approximation, this is where the Voronoi curves come in. As there are four grid points, there

will be $\binom{4}{2} = 6$ Voronoi curves for discriminating the pairs: **P** versus **Q** gives curve **P***v***Q**, **P** versus **R** gives curve **P***v***R**, **P** versus **S** gives **P***v***S**, **Q** versus **R** gives **Q***v***R**, **Q** versus **S** gives **Q***v***S**, and **R** versus **S** gives **R***v***S**. The equations of the curves **P***v***Q** is obtained from equation (3) and is given below while the rest of the Voronoi curve equations are listed in Appendix B.

P versus **Q** (cure **P***v***Q**)

$$\left((v-y)^2+(w-z)^2\right)\sin^2\theta_p - \left((v-(y+1))^2+(w-z)^2\right)\sin^2\theta_q = 0 \tag{8}$$

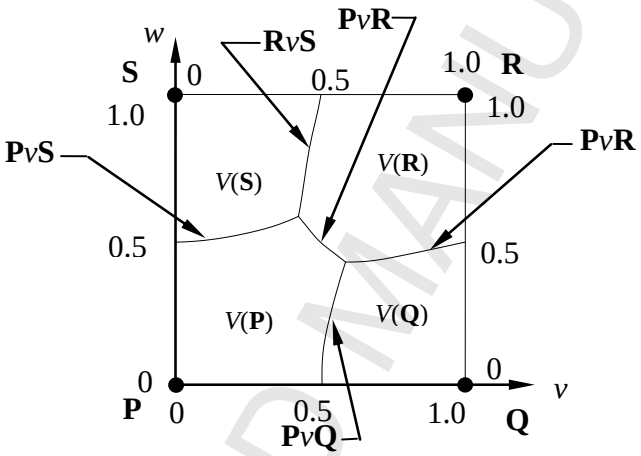
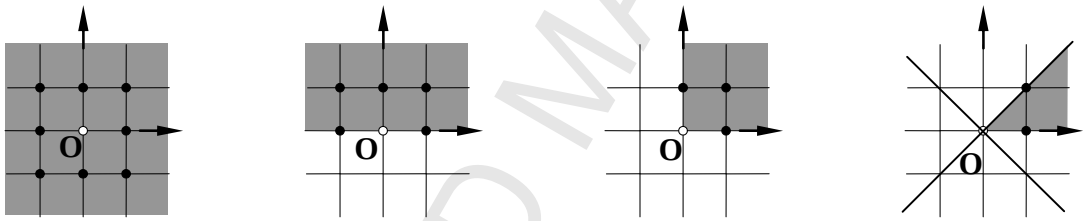


Figure 5. Voronoi curves on the x = 1 square **PQRS**

These Voronoi curves are expressed in terms of the local co-ordinates (v, w) on the square **PQRS**. Each curve consists of the terms x, y and z which are the grid co-ordinates of grid point $\mathbf{P}(x, y, z)$ as depicted in figure 5 with grid points shown as back dots. It is noted that the Voronoi curve **Q***v***S** does not contribute to any boundaries of the Voronoi cells. A grid point $\mathbf{D} \in \{\mathbf{P}, \mathbf{Q}, \mathbf{R}, \mathbf{S}\}$ is selected if the sample point **T** is a member of a Voronoi cell $V(\mathbf{D})$ or $\mathbf{T} \in V(\mathbf{D})$.

5. Symmetry in Three-Dimensional Space

It is useful to re-visit the Bresenham's Algorithm in two dimensions. Without exploiting symmetry, there are eight candidate grid points to select at the origin **O**. In figure 6(a), the white dot represents the grid point at the origin **O** while the eight candidate grid points are shown as black dots. Using a principal axis for a two-fold symmetry gives two half planes. Each half plane has five candidate grid points in order to reduce the approximation error to within ½ as depicted in figure 6(b). A four-fold symmetry is obtained by using two principal axes. The plane is partitioned into four quadrants with each offering three candidate grid points as shown in figure 6(c). Three lines such as two principal axes and one diagonal do not provide symmetry. Finally, four lines of symmetry (two principal axes and two diagonals) give an eight-fold symmetry. Each octant has two choices in Bresenham's Algorithm as illustrated in figure 6(d).



(a) No symmetry line (b) 1 symmetry line (c) 2 symmetry lines (d) 4 symmetry lines

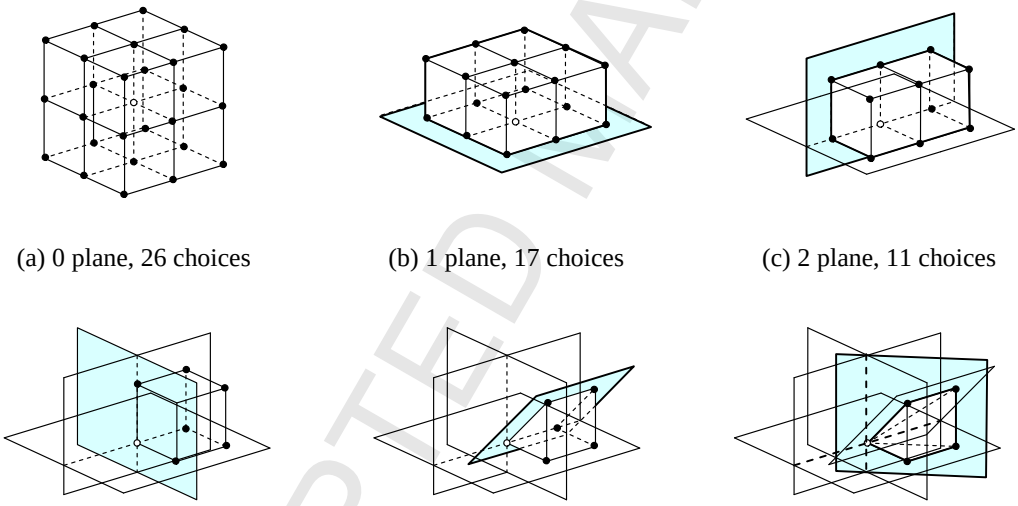
Figure 6. The number of candidate grid points with various symmetries applied.

Table 1 lists the number of symmetric lines, number of symmetry and number of grid point choices in each octant.

No. of lines of symmetry	No. of symmetries	choices in each octant
0	1	8
1	2	5
2	4	3
4	8	2

Table 1 Line symmetries in two-dimensional space.

A three-dimensional grid point has twenty six neighbours as choices as shown in figure 7(a). Using one principal plane gives two rectangular parallelepipeds. Each parallelepiped has nine choices plus eight that lie in the plane as seen in figure 7(b). Figure 7(c) depicts the symmetry resulting from using two principal planes. Six plus five or eleven choices arise. Eight-fold symmetry from three principal planes yields a smaller cube with seven choices to make as shown in figure 7(d). In addition to the principal planes, there are the "diagonal" planes. Slicing the small cube into two "wedges" gives six vertices each. Five of them are choices as shown in figure 7(e). Slicing the small cube with two "diagonal" planes gives a "four-sided pyramid" with five vertices. Four of which are candidates for choice as illustrated in figure 7(f).



(a) 0 plane, 26 choices (b) 1 plane, 17 choices (c) 2 plane, 11 choices

(d) 3 planes, 7 choices (e) 4 planes, 5 choices (f) 5 planes, 4 choices

Figure 7. Plane symmetries in three-dimensional space

Missing in figure 7 is the 64-fold symmetry, given as the last row of table 2. It turns out that there are four ways to construct a tetrahedron (half of a four-sided pyramid); this deserves some clarifications.

No. of planes	No. of symmetries	Choices in each partition
0	1	26 (as shown in figure 7(a))
1	2	17 (as shown in figure 7(b))
2	4	11 (as shown in figure 7(c))
3	8	7 (as shown in figure 7(d))
4	16	5 (as shown in figure 7(e))
5	32	4 (as shown in figure 7(f))
6	**64**	**3**

Table 2. Symmetries in three-dimensional space.

Geometrically, three grid points form a triangle. For instance, a four-sided pyramid shown in figure 7(f) is considered. Together with the origin **O** (shown as white dot in figure 8), the four points form a tetrahedron. But there are two ways to partition a square into two triangles as illustrated in figure 8.

(a) Through diagonal **SQ** with grid points **P**, **Q** and **S**

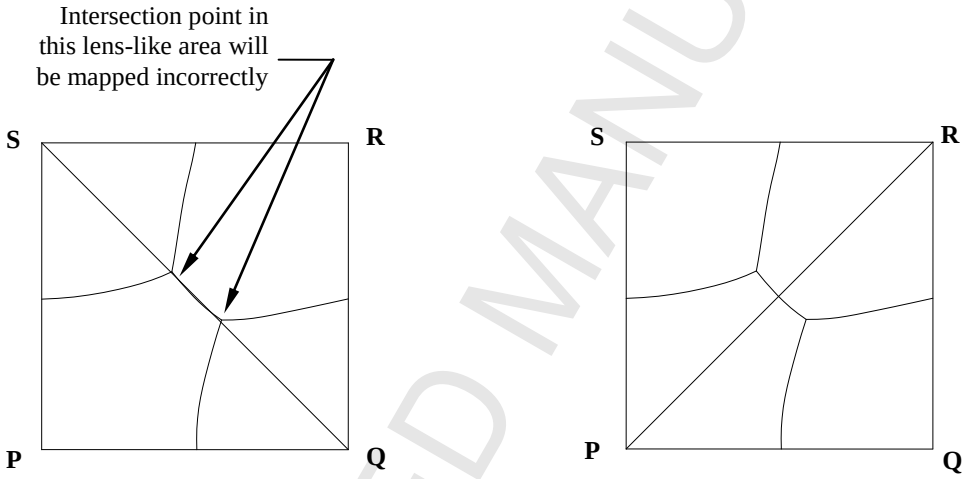
(b) Through diagonal **SQ** with grid points **Q**, **R** and **S**

(c) Through diagonal **PR** with grid points **P**, **S** and **R**

(d) Through diagonal **PR** with grid points **P**, **Q** and **R**

Figure 8. Two ways of partitioning the base of a tetrahedron.

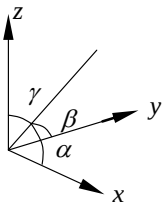
Now, when the Voronoi diagram in the square (with grid points **P**, **Q**, **R** and **S**) is taken into account, it can be seen that partitioning by the diagonal **SQ** is inferior to the diagonal **PR**. As shown in figure 9(a), there is a finite probability of resulting in an incorrect grid point selection. The intersection point (between the line segment and the square) might fall in the lens-like area between **SQ** and the Voronoi curve which will be mapped to grid point **R** incorrectly instead of grid point **P.** Therefore, the other diagonal **PR** as shown in figure 9(b) is chosen to avoid this situation. Such a selection is achieved by choosing the grid points with coordinates (x, y, z) and (x, y+1, z+1).



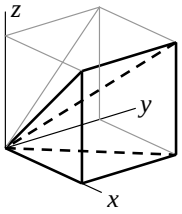
(a) Incorrect partition (b) Correct partition

Figure 9. Partitioning due to asymmetry

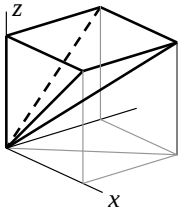
In fact, the positive rectangular parallelepiped ($x > 0$, $y > 0$, $z > 0$) in three-dimensional space (as shown in figure 7(d)) is partitioned into three four-sided pyramids. The selection of the four-sided pyramids depends on the orientation of a line which is represented by the direction cosine (α, β, γ). Each four-sided pyramid is characterized by the ranges of the direction cosine as shown in figure 10.



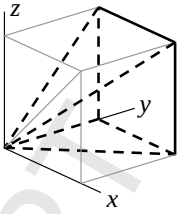
(a) direction cosine of a line



(b) $1 > \alpha > \frac{1}{\sqrt{3}}$, $\frac{1}{\sqrt{2}} > \beta > 0$, $\frac{1}{\sqrt{2}} > \gamma > 0$



(c) $\frac{1}{\sqrt{2}} > \alpha > 0$, $\frac{1}{\sqrt{2}} > \beta > 0$, $1 > \gamma > \frac{1}{\sqrt{3}}$



(d) $\frac{1}{\sqrt{2}} > \alpha > 0$, $1 > \beta > \frac{1}{\sqrt{3}}$, $\frac{1}{\sqrt{2}} > \gamma > 0$

Figure 10. Partitioning of the rectangular parallelepiped

Consequently, the grid point selection to partition the square (into two triangles) is four-sided pyramid dependent. The grid point pair (x, y, z), (x+1, y+1, z) and (x, y, z), (x+1, y, z+1) should be selected for the four-sided pyramids shown in figure 10(c) and 10(d) respectively.

Bresenham's algorithm for a two-dimensional line adopts an eight-fold symmetry in the plane and there are two choices to make in each octant. A three-dimensional version involves sixty-four-fold symmetry with three choices in each tetrahedron. From two-dimensional to three-dimensional, a unit length becomes a unit square and the number of symmetry lines doubles from three to six symmetry planes, while the number of symmetries increases quadratically from eight to sixty four. Since making a selection involves computing the distance between the candidate grid point and the given line segment, it stands to reason that such computations should be minimized. It follows that the Bresenham's algorithm for a three-dimensional line should also minimize the number of selections which is the case of three choices in a 64-fold symmetry as listed in the last row of table 2.

6. The Algorithm

The original approach[1] of Bresenham's algorithm for plotting a two-dimensional line between grid origin (0, 0) to a grid point (a, b) is adopted to present the three-dimensional

extension of the algorithm. Any line which does not originate from the grid origin needs to do a transformation. Given that the line segment goes from grid point **A** (0, 0, 0) to point **B** (*a*, *b*, *c*) in the four-sided pyramid shown in figure 10(b), with $a \geq b$ and $a \geq c$, there are *a* iterations.

```
    Procedure 3DBresenhamLine { A (0, 0, 0), B ( a, b, c) }
        Output A
        for j = 1 to a do begin
Step 1      T(u, v, w) = Intersect {line AB, plane x = j}
Step 2      Identify the square PQRS of consecutive grid points on the plane x = j
            that contains T
Step 3      Construct two triangles and identify the one containing T
Step 4      if T is in ΔPQR then
            Output {P, Q or R} based on the membership of T in the Voronoi diagram
Step 5      else
            Output {P, R or S} based on the membership of T in the Voronoi diagram
        end do
    EndProcdure
```

In Step 1, line-plane intersection can be done with multiplication and division, as **T** (*u*, *v*, *w*) = **T** $\left(j, j \times \frac{b}{a}, j \times \frac{c}{a} \right)$. In Step 2, the coordinates of the corners of the square are found from the point of intersection by rounding: $x = j$, $y = \lfloor v \rfloor$ and $z = \lfloor w \rfloor$. Step 3 involves the construction of a diagonal plane ($v - w - y + z = 0$) through **PR** for determining in which side **T** lies (refer to figure 8(c) and 8(d)). Step 4 and 5 decide which grid point will be selected based on the membership of **T.**

Figure 11 shows the partitions with the Voronoi curves as the partition boundaries. The conditions for having the sample point **T** in the partition are listed in the figure.

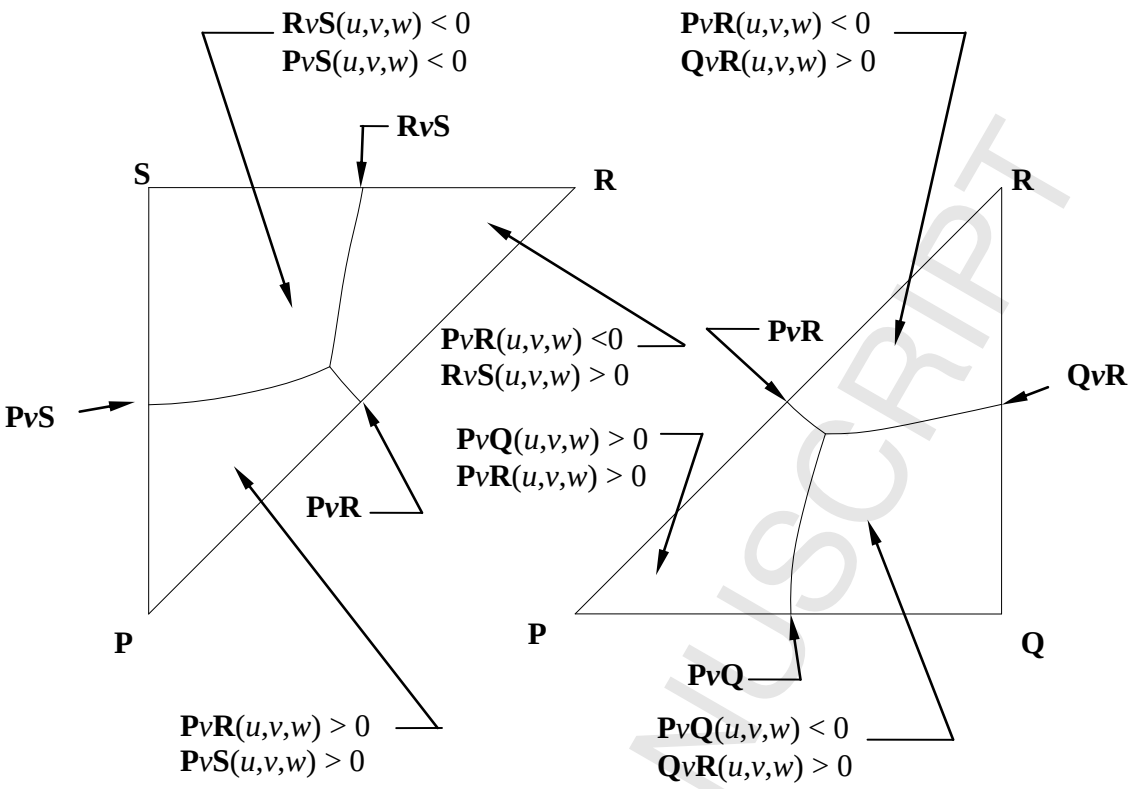


Figure 11. Partitioning of the triangle **PQR** and **PRS**

For illustration, the function **P***v***Q** for discriminating the two grid points **P**(*x*, *y*, *z*) and **Q**(*x*, *y*+1, *z*+1) returns a numerical value: greater than zero, equal to zero and smaller than zero.

Function **P***v***Q** (*u*, *v*, *w*)

return $\left(\left((v-y)^2+(w-z)^2\right)\sin^2\theta_p-\left((v-(y+1))^2+(w-z)^2\right)\sin^2\theta_q\right)$

End - function

The details for the three-dimensional Bresenham line algorithm are ready:

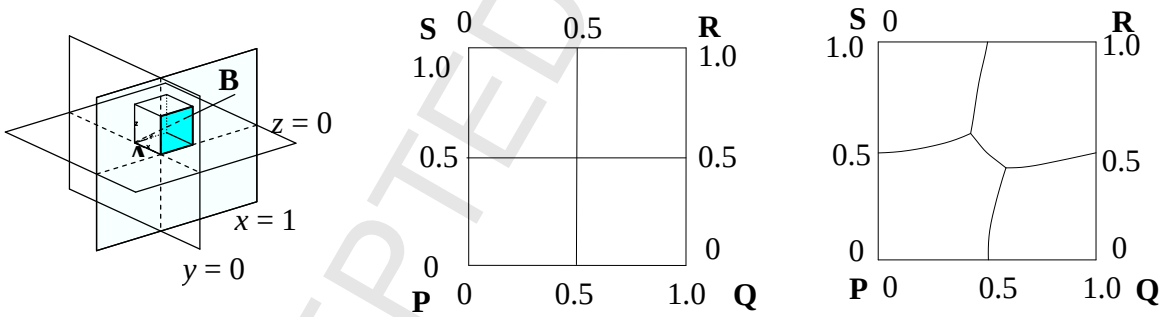
```
Procedure 3DBresenhamLine { A (0, 0, 0), B ( a, b, c) }
        Output A
        for j = 1 to a do begin
Step 1      T(u, v, i) = T ( j, j × b/a, j × c/a)
Step 2      x = j; y =  ⌊v⌋;   z = ⌊w⌋;
Step 3 & 4  if (v - w   - y + z) < 0, then
                if PvQ(u,v,w) > 0 and PvR(u,v,w) > 0 then choose P
                if PvQ(u,v,w) < 0 and QvR(u,v,w) > 0 then choose Q
                choose R
Step 5      else
                if PvR(u,v,w) > 0 and PvS(u,v,w) > 0 then choose P
                if PvR(u,v,w) < 0 and RvS(u,v,w) > 0 then choose R
                choose S
        end-do
End-Procedure
```

For the line in the other two four-sided pyramids, the algorithm can be modified by a cyclic rotation of the *x*, *y*, *z* coordinate. For instance, if the line is in the four-sided pyramid shown in figure 10(c), the *x*, *y*, *z* coordinates will be rotated to *z*, *x*, *y* coordinates. And the sampling is performed along the *z* axis.

7. Discussion

It is necessary to inquire if the Bresenham's algorithm can be extended to three dimensions since the line representation in a discrete three-dimensional space arises in many engineering applications. A reasonable place to start is to employ a "squaring" heuristic. If the two-dimensional selection is formulated as a comparison between the *y*-intercept of the given line with the value of $\frac{1}{2}$ (in the *y*-axis), then the three-dimensional selection is one on a square of unit area (involving both *x* and *y*). Reasoning suggests that the number of choices should also be "squared". The two-dimensional selection in each octant has two choices; the three-dimensional problem should have four candidates. A simple algorithm for candidate (grid point) selection is to obtain the grid point (among four candidates) which has the

shortest perpendicular distance to the line. Hence, there are totally four distance computations and six comparisons for every sample point. The proposed algorithm suggests a more efficient implementation involves 3 grid points for choosing. Of the 3 grid points, there must be pair-wise discriminations, involving $\binom{3}{2}$ = 3 Voronoi curves. The Voronoi curve (equation (8) for instance) is actually a comparison of two squared perpendicular distances from two grid points. The elimination of the square roots in the expression (compared with direct perpendicular distance computation and comparison) simplifies the computation a bit. More importantly, only three distance computations and three comparisons are involve in the worst case (and three distance computations and two comparisons are involved in the best case). As a result, the proposed algorithm is more efficient than direct distance computation and comparison by setting up this simple hierarchy (four candidates are separated into two groups and each group consists of three candidates), especially when there are huge number of sample points.



(a) A line segment intersects the square in the plane $x = 1$.

(b) Four selection regions in the plane $x = 1$ with angle of intersection equal to 90^0.

(c) Four selection regions in the plane $x = 1$ with angle of intersection not equal to 90^0

Figure 12. Nearest neighbor as the 3D selection

Figure 12(a) shows a line segment beginning at A = (0, 0, 0) "piercing" a unit square, in the plane $x = 1$, with coordinates for its four corners **P**, **Q**, **R** and **S** at (1, 0, 0), (1, 0, 1), (1, 1, 1),

and (1, 1, 0). If the given line segment intersects a plane at exactly 90^0, the partitioning of the square looks like a "cross", as shown in figure 12(b). This is the special three-dimensional case in which the two-dimensional idea extends directly. In general, the partitions are bounded by algebraic curves as shown in figure 12(c).

The Voronoi diagram for the “half and half” approach which projects the sample point onto the edges of the square **PQRS** to determine the grid point is shown in figure 13. The Voronoi cells for each grid point are constant and identical. The Voronoi cells containing the sample point are coloured as white in the figure.

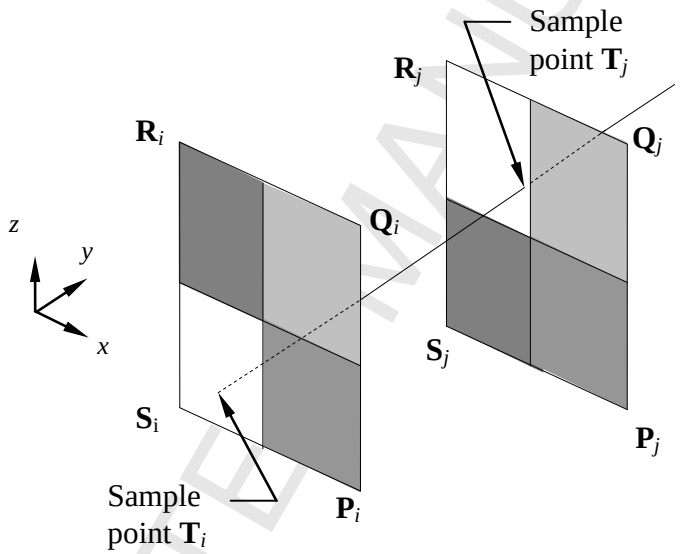
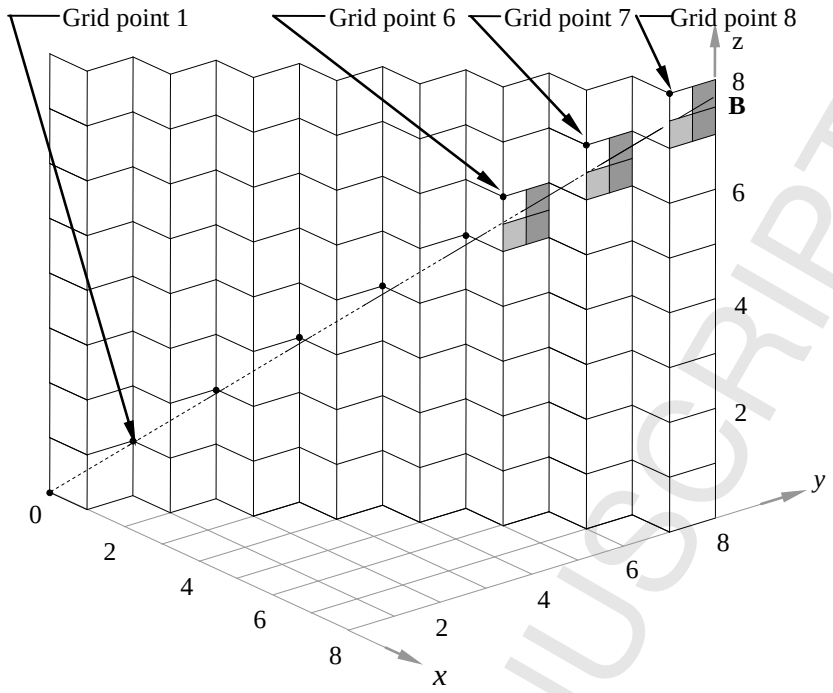


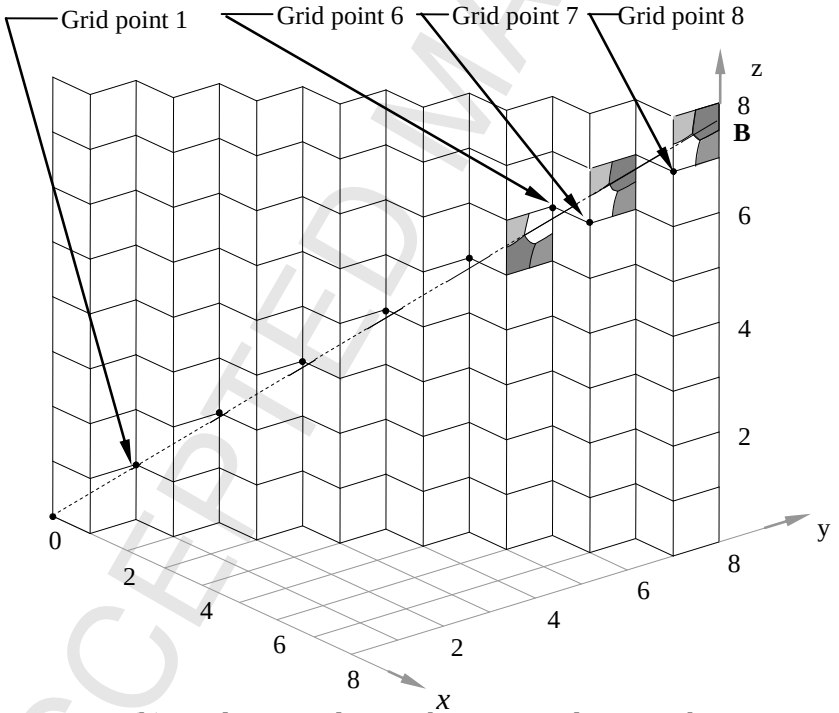
Figure 13. The Voronoi diagram for grid point selection in “half and half” approach for 3D Bresenham algorithm

However, such an approach does not give an accurate line representation in three-dimensional space. Figure 14 shows the difference between two approaches for representing a line from **A**(0, 0, 0) to **B**(80, 72, 75). Figure 14(a) shows the first eight grid

points (black dots) based on the “half and half” heuristic while figure 14(b) shows the line representation by the proposed three-dimensional Bresenham algorithm.



(a) Grid point selection by "half and half" approach



(b) Grid point selection by proposed approach

Figure 14. Line representation in three-dimensional space.

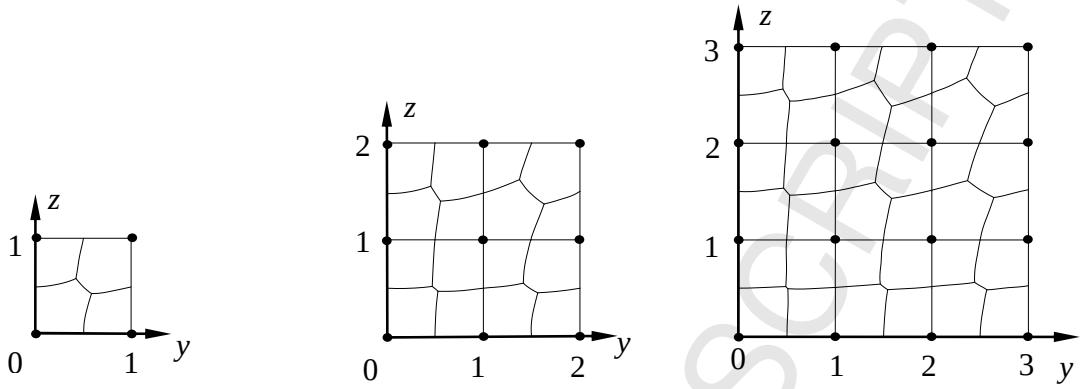
The line is sampled and the grid points are selected. The coordinates and distances from the first eight grid points to the line are listed in table 3.

	Sample point	Half and half approach		Proposed approach	
		Grid point	Distance from grid point to line	Grid point	Distance from grid point to line
1	(1, 0.9, 0.93	(1, 1, 1)	0.08	(1, 1, 1)	0.08
2	(2, 1.8 1.88)	(2, 2, 2)	0.15	(2, 2, 2)	0.15
3	(3, 2.7, 2.81)	(3, 3, 3)	0.23	(3, 3, 3)	0.23
4	(4, 3.6, 3.75)	(4, 4, 4)	0.30	(4, 4, 4)	0.30
5	(5, 4.5, 4.69)	(5, 5, 5)	0.58	(5, 5, 5)	0.58
6	(6, 5.4, 5.63)	(6, 5, 6)	0.55	(6, 6, 6)	0.45
7	(7, 6.3, 6.56)	(7, 6, 7)	0.52	(7, 6, 6)	0.41
8	(8, 7.2, 7.50)	(8, 7, 8)	0.51	(8, 7, 7)	0.37

Table 3. The coordinates of grid points and sample points of a line based on two three-dimensional Bresenham algorithms

It can be seen that both approaches select the same grid points (these are grid point 1 to 5 in this example) initially. The selection (staring from grid point 6 in the example) differs as the sample points are further away from the line origin **O**. Clearly, the proposed algorithm selects the grid points which are closer to the line than that of the "half and half" approach. In fact, the equations of the Voronoi curves explain the reason. Equations (8), (B1) to (B5) are the mathematical forms of the Voronoi curves in the square $\mathbf{P}_i\mathbf{Q}_i\mathbf{R}_i\mathbf{S}_i$ with the coordinate of grid point $\mathbf{P}_i$ as (x_i, y_i, z_i). The dependence of the Voronoi curves on the *x*, *y* and *z* -coordinate are shown in figure 15. Figure 15(a), 15(b) and 15(c) show the Voronoi cells with $x = 1$, $x = 2$ and $x = 3$ respectively. The shapes of the Voronoi cells are not constant and non-identical. The progression of 1×1, 2×2, and 3×3 squares is to suggest that the deviation of the Voronoi cell shape from a square increases with the length of the line. Therefore, if the

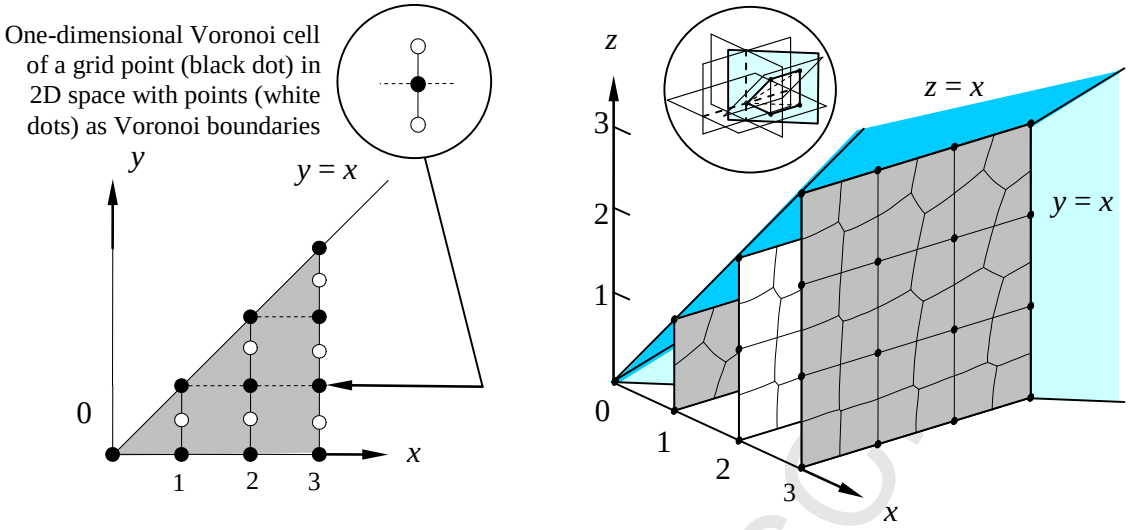
sample point is close to the centre of square **PQRS**, the "half and half" approach will most likely choose an inappropriate grid point (the selected grid point is not the closest to the line).



(a) Voronoi cells in $x = 1$ plane (b) Voronoi cells in the $x = 2$ plane (c) Voronoi cells in the $x = 3$ plane

Figure 15. Voronoi diagram among the grid point at $x = 1$, $x = 2$ and $x = 3$

Figure 16 shows the arrangement of the Voronoi cells in two-dimensional and three-dimensional space. Figure 16(a) shows the Voronoi cells of grid points at $x = 1$, $x = 2$ and $x = 3$ in two-dimensional space. The Voronoi cells are vertical lines (which is one dimensional) with mid-points between two adjacent grid points as Voronoi boundaries (which is zero dimension) as shown in the circle diagram. Figure 16(b) shows the three-dimensional situation. The circle diagram in figure 16(b) contains a "four-sided pyramid" from figure 7(f) for comparison. The Voronoi curves approach a "cross", as the angle of intersection approaches 90^0, as in the lower-left square in the figure.



(a) Voronoi cells of each grid point with x = 1, 2 and 3 in two-dimensional space.

(b) Voronoi cells of each grid point with x = 1, 2 and 3 in three-dimensional space.

Figure 16. Voronoi cells for line representation in two-dimensional and three-dimensional space.

It may be said that the Voronoi cell $V(\mathbf{D})$ has different distance *metrics* $\begin{bmatrix} \sin\theta_D & 0 \\ 0 & \sin\theta_D \end{bmatrix}$ ($\forall \mathbf{D} \in \{\mathbf{P}, \mathbf{Q}, \mathbf{R}, \mathbf{S}\}$) for measuring distances to the given line segment. Re-consider the grid point **P** in figure 4. Since the line **PP**' is not in the square **PQRS**, the apparent distance measured from **P** to the intersection point is distorted due to the projection of the length **PP**' on to the plane of the square. The amount of distortion is different for the other grid points owing to their different locations, hence different distance metrics and non-identical Voronoi cells arise.

It is comforting to note, however, that four of the five curves terminate on the boundary of the unit square at exactly $\frac{1}{2}$ the distance between the two adjacent grid points. In other words, the three-dimensional selection (involving Voronoi cells in a unit square) degenerates to a two-dimensional selection (involving mid-point comparison in the unit interval), but the

two-dimensional technique does not extend readily to three-dimensions. Hence, the "half and half" approach does not really select the right set of grid points, which should possess the minimum distances from the line, to represent a line in three dimensions.

Summary

The generalization of Bresenham's algorithm to three-dimensional has been manifested. A Voronoi diagram is employed for grid point selection. The proposed algorithm yields a more accurate grid point representation for a three-dimensional line. Similar to the Bresenham's algorithm, the three-dimensional algorithm also makes use of the symmetry to raise the computation efficiency. In addition, the computation efficiency is further improved by setting up a simple hierarchy to reduce the number of distance computation and comparison. However, the deployment of integer arithmetic does not seem encouraging due to the algebraic Voronoi curves.

References

1. Bresenham JE. Algorithm for computer control of a digital plotter. IBM Systems Journal. 1965; 4(1):25-30.
2. Dimitrov LI, Sramek M. Using 3D-Bresenham for resampling structured grids. Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization and Transmission. 2004. p.962-930.
3. Proctor MD, Gerber WJ. Line-of-sight attributes for a generalized application program interface. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology. 2004; 1(1):43-57.
4. Liu XW, Cheng K. Three-dimensional extension of Bresenham's algorithm and its application in straight-line interpolation. Proceedings of the Institution of Mechanical Engineers. 2002; 3:459-463.
5. Hsu Wm. Segmented ray casting for data parallel volume rendering. Proceedings of the 1993 Symposium on Parallel Rendering. 1993. p.7-13.
6. Schroder P, Gordon S. Data parallel volume rendering as line drawing. Proceedings of the 1992 Workshop on Volume Visualization. 1992. P.25-32.
7. Gray M, Fairfield N, Stone W, Wettergreen D, Kantor G, Sharp JM. 3D mapping and characterization of sistema zacatón from DEPTHX (DEep Phreatic THermal eXplorer). Proceedings of KARST08. 2008.
8. Sanchiz JM, Fisher RB. A next-best-view algorithm for 3D scene recovery with 5 degrees of freedom. Proceedings of British Machine Vision Conference. 1999. P.163-172.

9. Orman N, Kim H, Sakamoto R, Toriyama T, Kogure K, Lindeman R. GPU-based optimization of a free-viewpoint video system. Electronic Letters on Computer Vision and Image Analysis. 2008; 7(2):121-133.
10. Scherer S, Singh S, Chamberlain L, Elgersma M. Flying fast and low among obstacles: methodology and experiments. The International journal of Robotics Research. 2009; 7:549-574.
11. Paiva A, Petronette F, Lewiner T, Tavares G. Particle-based viscoplastic fluid/solid simulation. Computer-aided Design. 2009; 41(4):306-314.
12. Foley J D, Van Dam A, Feiner S and Hughes J F. Computer Graphics: Principles and practice in C (2nd Editions), Addison-Wesley,1997
13. Au CK, Spatial and temporal competition as a two dimensional kinetic Voronoi diagram. Computer-aided Design, 2008; 40(2):139-149.

Appendix A

The expressions of the angle θ_q, θ_r and θ_s are

$$\cos\theta_q = \frac{v(y+1-v)+w(z-w)}{\sqrt{(x^2+v^2+w^2)\left((y+1-v)^2+(z-w)^2\right)}} \quad \text{(A1)}$$

$$\cos\theta_r = \frac{v(y+1-v)+w(z+1-w)}{\sqrt{(x^2+v^2+w^2)\left((y+1-v)^2+(z+1-w)^2\right)}} \quad \text{(A2)}$$

$$\cos\theta_s = \frac{v(y-v)+w(z+1-w)}{\sqrt{(x^2+v^2+w^2)\left((y-v)^2+(z+1-w)^2\right)}} \quad \text{(A3)}$$

Appendix B

The equations of the curves are listed:

P versus **R** (curve **P***v***R**)

$$\left((v-y)^2+(w-z)^2\right)\sin^2\theta_p - \left((v-(y+1))^2+(w-(z+1))^2\right)\sin^2\theta_r = 0 \quad \text{(B1)}$$

P versus **S** (curve **P***v***S**)

$$\left((v-y)^2+(w-z)^2\right)\sin^2\theta_p - \left((v-y)^2+(w-(z+1))^2\right)\sin^2\theta_s = 0 \quad \text{(B2)}$$

Q versus **R** (curve **Q***v***R**)

$$\left((v-(y+1))^2+(w-z)^2\right)\sin^2\theta_q - \left((v-(y+1))^2+(w-(z+1))^2\right)\sin^2\theta_r = 0 \quad \text{(B3)}$$

Q versus **S** (curve **Q***v***S**)

$$\left((v-(y+1))^2+(w-z)^2\right)\sin^2\theta_q-\left((v-y)^2+(w-(z+1))^2\right)\sin^2\theta_s=0 \qquad \text{(B4)}$$

S versus **R** (curve **S***v***R**)

$$\left((v-(y+1))^2+(w-(z+1))^2\right)\sin^2\theta_s-\left((v-y)^2+(w-(z+1))^2\right)\sin^2\theta_r=0 \qquad \text{(B5)}$$